Week 5: Assignment

1. Which of the following are true for a test bench?
    a. Both $display and $monitor can be used to print the current value of variables as the statement encountered.
    b. The inputs to the design under test must always be declared as "reg" type whereas outputs will be "wire" type.
    c. The synthesis tool requires test bench for synthesizing the module.
    d. Both "initial" and "always" procedural blocks can be used inside a test bench.

    Answer: (b) and (d)

    The $display directive prints immediate values of text or given list of variables on screen whereas $monitor directive only prints when there is a change of values of some variables form the given list. Thus option (a) is false. The inputs and outputs of the designed module must be declared as "reg" type and "wire" type inside the test bench module. Thus option (b) is true. Test bench is only required during simulation and it is not needed when synthesizing the module. Thus option (c) is false. Test bench uses both initial and always block for providing test inputs. Thus option (d) is also true.

2. Which of the following is true for an "always" block?
    a. It can only be used to write synthesizable code.
    b. Only blocking and non-blocking assignments are allowed inside the block.
    c. The right hand side of the assignments can only be wire type variables inside the block.
    d. Continuous assignment is allowed inside the block.

    Answer: (b)

    The "always" block also used inside test bench for generating inputs. Thus option (a) is false. Both blocking and non-blocking assignments are allowed inside "always" block. Thus option (b) is true. The RHS of an assignment inside "always" block can be register or wire type. Thus option (c) is false. Only procedural assignment statements are allowed inside the "always" block. Thus option (d) is also false.

3. Which of the following statements are true?
    a. Both $dumpall and $dumpvars with no parameters can be used interchangeably to dump all variables.
    b. The dumped file using $dumpfile contains information of all variables declared inside the testbench.
    c. The $dumpon is used to start dumping values of all variables in dump file after specifying dump file name.
    d. The $dumpoff statement forces dumping all variables with "x" values and stop dumping further change of variables

    Answer: (d)

The $dumpall directive dump current values of the variables, whereas $dumpvar directive without any parameter dump all values of the all variables only when there is change in values. Thus option (a) is false. The dumped file contains information about changes of values of selected variables. Thus option (b) is false. The $dumpon directive is used to start previously stopped dumping of variables. Thus option (c) is false. After encountering the $dumpoff statement, all variables are dumped with "x" values and next change of variables will not be dumped. Thus option (d) is true.

4. Which of the following are true about the following verilog directive

```
`timescale 10ns / 10ps
```

   a. The synthesis tool interprets 10 nanoseconds as unit of delay.
   b. The directive is only considered during simulation and ignores during actual synthesis.
   c. The simulator interprets 10 nanoseconds as unit of delay with a resolution of 10 picoseconds.
   d. The directive is considered during simulation and synthesis both.

Answer: (b) and (c)

The timescale directive is only considered during simulation, it is ignored by synthesis tool. Thus option (a) is false and option (b) is true. The simulator interprets 10ns as unit of delay with precision 10ps. Thus option (c) is true and option (d) is false.

5. Select the appropriate interpretation of the following verilog code

```
`timescale 10ns / 100ps
module test_dut;
reg [2:0] x;
initial begin #10.4567;
      x = 4;
end
endmodule
```

   a. The value of x is initialized after a delay of 100 ns.
   b. The value of x is initialized after a delay of 104.600 ns.
   c. The value of x is initialized after a delay of 104 ns.
   d. The value of x is initialized after a delay of 105 ns.

Answer: (b)

Here timescale is 10 ns. Thus the delay of 10.4567 will become 104.567 ns and the fractional part will be rounded to the nearest time precision i.e. 10.600 ns. Thus option (b) is correct and option (a), (c) and (d) are not correct.

6. What does the following code segment do?

```
reg [3:0] x;
initial
      clk = 1'b0;
always #10 clk = ~clk;
initial begin #5;
      for (i=0; i<16; i++)
            #20 x = i;
```

```
end
```

a. The vector "x" will be assigned values 0 to 15 at a regular interval of 20 time unit before the raising edge of the clock "clk".

b. The vector "x" will be assigned values 0 to 15 at a regular interval of 20 time unit before the failing edge of the clock "clk".

c. The vector "x" will be assigned values 0 to 15 at each clock edge.

d. None of the above.

Answer: (a)

Here the "clk" variable is initialized with 0 and changes it's state at a regular interval of 10 time unit. So the raising edge of the clk variable will be at time unit 10, 30, 50, 70, … . The for loop after an initial delay of 5 time unit and variable "x" is assigned values 0 to 15 after a delay of 20 time unit. Thus value of "x" will be updated at time units 25, 45, 65, …, so on before the raising edge of the clock "clk".

7. What will be the time period of the clock (**clk**) generated by the following code segment?

```
`timescale 10ns/1ns
module test_dut;
reg clk
initial
      #0.35 clk = 1'b0;
always
      #0.63 clk = ~clk;
endmodule
```

a. 6.2 ns

b. 7 ns

c. 12 ns

d. 9.7 ns

Answer: (c)

Here the time scale is 10ns with precision 1ns. Thus the "clk" variable will be toggled after 6 ns and the "clk" value will be repeated after 12ns. Thus option (c) is correct, and options (a), (b) and (d) are incorrect.

8. What will be the time period of the clock (**clk**) generated by the following code segment?

```
`timescale 10ns/10ns
module test_dut;
initial
clk = 1'b1;
always
forever  begin
    #2.2 clk = 1'b0;
    #1.5 clk = 1'b1;
end
endmodule
```

a. 40 ns

b. 3.7 ns
c. 37 ns
d. 30 ns

Answer: (a)

Here the time unit is 10ns with precision 10ns. Thus here the clock will remain 0 for 20 time unit and 1 for 20 time unit. Thus option (a) is correct, and options (b), (c) and (d) are incorrect.

9. Which of the following statements is/are true?
   a. For both Mealy and Moore machines the present state and present input determines the next state.
   b. In Mealy machine the output is associated with present state.
   c. In More machine the output is associated with present state and present input.
   d. Both Mealy and Moore machines are used to realize sequential circuit.

   Answer: (a) and (d)

   The next state is determined as a function of present state and present input for both Mealy and Moore machines. Thus option (a) is true. The output of Mealy machine is associated with present input and present state whereas output of More machine depends only on present state. Thus option (b) and (c) are false. Both state machines are used for realizing sequential circuit. Thus option (d) is also true.

10. The number of flip-flops will be realized by the following code segment will be .....

```
module demo_fsm (clk, signals);
input clk;
output reg [4:0]  signals;
parameter  S0=0, S1=1, S2=2; S3=3, S4=4;
reg [0:2]  state;
always  @(posedge clk)
case (state)
      S0: begin
           signals<= 5'b11100;  state <= S3;
           end
      S1: begin
           signals<= 5'b10010;  state <= S2;
      end
      S2: begin
           signals<= 5'b01011;  state <= S0;
      end
      S3: begin
           signals<= 5'b01001;  state <= S4;
      end
      S4: begin
           signals<= 5'b01101;  state <= S1;
      end
      default: begin
           signals<= 5'b01010;  state <= S0;
```

```
        end
    endcase
    endmodule
```

Answer: 8

Here flip-flops will be synthesized for both the **signals** and **state** variables. Thus the number of flip-flop will be 5 + 3 = 8.